



SOLVING JOB SHOP SCHEDULING PROBLEMS WITH CONSULTANT GUIDED SEARCH METAHEURISTICS

D.Deepanandhini¹ T.Amudha²
M.Phil. Research Scholar¹ Assistant Professor²

Department of Computer Applications
Bharathiar University, Coimbatore – 641046, Tamilnadu, India.

Abstract: Nature Inspired Computing (NIC) is the research area that aims to get ideas by observing how nature behaves in various situations to solve problems. NIC when compared to the traditional computing systems, respond slowly but produces good results. Nature-Inspired algorithms is a class of algorithms that mimic the problem solving behavior from nature. In this paper Job Shop Scheduling Problem (JSSP) is solved by using the Consultant Guided Search algorithm (CGS) a Nature-Inspired metaheuristic algorithm. JSSP instances from OR library are selected and solved by implementing CGS algorithm. It was observed that CGS algorithm has obtained best-so-far solutions for almost all the instances taken for study.

Keywords: Nature Inspired Computing, Job Shop Scheduling Problem, Consultant Guided Search.

I. Introduction

Nature-Inspired algorithms are a class of algorithms that imitate the way nature performs. Nature provides inspiration for countless researchers in many ways. Bio-Inspired Computing (BIC) is a subset of Nature Inspired Computing. BIC is an upcoming technique for solving NP-hard problem, and NP-complete problems. The behavior of social insects and characteristics are playing main role to get better solutions. Airplanes have been designed based on the structures of birds' wings. Robots have been designed in order to imitate the movements of insects. The metaheuristic can be a general algorithmic framework; it can be applied to different Combinatorial Optimization (CO) problems with relatively some modifications to make them adapted to a specific problem [7], [3], [4].

Combinatorial Optimization (CO) problems involve finding values for discrete variables such that the optimal solution with respect to a given objective function is found. A CO problem is either maximization or minimization problem which is associated with a set of problem instances [25]. CO problems are usually called as search problems [20], [17]. Examples of real-world combinatorial optimization problems are the shortest-path problems, minimum cost plan problem to deliver goods to customers, optimal assignment of employees to tasks, routing scheme for data packets in the Internet, optimal sequence of jobs to be processed in a production line, and many more. For combinatorial optimization problems, all the algorithms could not come up with optimal solution with in polynomial time. JSSP is one of the Combinatorial Optimization (CO) problems [19]. These problems are resolved using nature inspired techniques.

Non-deterministic Polynomial time hard problem is called as NP-Hard problem. Example of an NP-Hard problem is the optimization problem of finding the least-cost cyclic route through all nodes of a weighted graph [25]. This is commonly known as the Travelling Salesman Problem (TSP). A problem is NP-Hard if solving it in polynomial time would make it possible to solve all problems in class NP in polynomial time [25].

Nature inspired algorithms such as ACO (Ant Colony Optimization) [2], [12], [14], BCO(Bee Colony Optimization)[5], [18], PSO(Particle Swarm Optimization)[26], [16], GA (Genetic Algorithm)[15], SA (Simulated Annealing) [23], SS (Scatter Search)[1] and TS (Tabu Search)[6], [13] and applied to the JSSP. In this paper the new CGS metaheuristics algorithm is implemented and tested to solve JSSP instances. Scheduling for job shop is very important in both fields of production management and combinatorial optimization. Scheduling is an optimization problem that exists almost everywhere in real world.

II. Related Work

Shahrzad Nikghadam et al., (2011), have introduced minimizing earliness and tardiness costs in Job Shop Scheduling Problem (JSSP) considering job due dates. The objective function is to minimize earliness and tardiness costs which consists of storage costs, backorder costs, and lost sales cost, which means that, machines can be used more efficiently and moreover, tasks can be allocated properly. This scheduling not only focuses on finding optimal or closer to optimal solution for JSSP but also minimizes the cost [27].

Serban Iordache (2010) introduced Consultant-Guided Search (CGS), a new metaheuristic for combinatorial optimization problems, based on the direct exchange of information between individuals in a population. It exemplifies the application of this metaheuristic to a specific class of problems by introducing the CGS-TSP algorithm, an instantiation of Consultant-Guided Search for the Traveling Salesman Problem (TSP). To determine whether direct communication approach can be made to compete with stigmergy-based methods, he compared the performance of CGS-TSP with that of Ant Colony Optimization algorithms. Its experimental results proved that the solution quality obtained by CGS-TSP is highly comparable with or in some cases better than that obtained by Ant Colony System and MAX-MIN Ant System [10].

Hashni.T and Amudha.T (2012) have applied hybrid nature-inspired algorithm for solving Quadratic Assignment Problem (QAP). Consultant Guided Search (CGS) and Genetic Algorithm (GA) are some of the nature-inspired metaheuristic algorithms inspired from nature. They hybridized CGS with GA and a new method CGS-GA was proposed for solving the benchmark instances of QAP. The ability of the proposed CGS-GA was investigated through the performance of quite a few runs of the algorithm on well-known test problems. The results obtained by the proposed CGS-GA for QAP are much better when compared to the result obtained by CGS algorithm. They suggested that CGS-GA algorithm can be modified and be made suitable for other types of combinatorial optimization problems such as Vehicle Routing, Scheduling, Travelling Salesman Problem and Bin Packing etc., [8].

III. Solving JSSP using CGS Algorithm

The objectives of this research work are

- To solve Job Shop Scheduling Problem (JSSP) by using a relatively new Nature-Inspired technique, Consultant Guided Search (CGS).
- To examine the efficiency of CGS in solving benchmark instances of Job Shop Scheduling Problem (JSSP).

A. Job Shop Scheduling Problem (JSSP)

Job Shop Scheduling Problem (JSSP) can be formulated as follows. There is a set of N jobs $J = (J_1, J_2, J_3, J_4, J_5, \dots, J_n)$ and a set of M machines $M = (M_1, M_2, M_3, M_4, \dots, M_m)$. The jobs consist of several operations. Each job is routed through the M machines in a pre-defined order, which is also known as operation precedence constraints. The processing of a job on one machine is called an operation, and the processing of job i on machine j is denoted by u_{ij} . So the set of operations can be defined as $O = \{u_{ij} \mid i \in [1, n], j \in [1, m]\}$, in which n denotes the number of jobs and m denotes the number of machines. Once processing is initiated, an operation cannot be interrupted, and concurrency is not allowed. That is u_{ij} cannot begin processing until $u_{i,j-1}$ has completed. Each operation of $u_{ij} \in O$ on machine j of job i must have an integral processing time p_{ij} ($p_{ij} > 0$), which is also known as the machine processing constraints. The set O is decomposed into chains corresponding to the jobs: if the relation $u_{ip} \rightarrow u_{iq}$ is existed in a chain, then both operations u_{ip} and u_{iq} belong to job J_i and there is no machine k , which is not p or q , and relations such as $u_{ip} \rightarrow u_{ik}$ or $u_{ik} \rightarrow u_{iq}$. This means, in relation $u_{ip} \rightarrow u_{iq}$, u_{iq} is directly immediate to u_{ip} .

The JSSP has two constraints, known as the operation precedence constraint and machine processing constraint:

- 1) The operation precedence constraint on the job is that the order of operations of job is fixed and the processing of an operation cannot be interrupted and concurrent.
- 2) The machine processing constraint is that only a single job can be processed at the same time on the same machine.

B. Consultant Guided Search (CGS) Algorithm

Consultant-Guided Search (CGS) metaheuristic is a swarm intelligence technique for solving hard combinatorial optimization problems, which it takes inspiration from the way real people make decisions based on advice received from consultants [9]. An entity of the CGS population is a virtual person, which can concurrently act both as a client and as a consultant. As a client, a virtual person constructs solution to the problem at each iteration. As a consultant, a virtual person provides advice to clients. Usually at each step a solution is to be constructed. At the beginning of each iteration, a client chooses a consultant based on its **individual preference** and on the consultant's **reputation**. The reputation of a consultant increases with the number of successes achieved by its clients. A client achieves a **success**, if it constructs a solution better than all solutions established until that point by any client guided by the same consultant. If the consultant's reputation sinks below a minimum value, it will take a **sabbatical leave**, during which it will stop offering advice to clients and it will instead start searching for a new strategy to use in the future [24]. In the following CGS metaheuristic pseudo

code, a virtual person may be in any one of the following state: *normal* and *sabbatical*. The consultant used to guide the solution construction is chosen based on its reputation and on the personal preference of the client. Since a virtual person act concurrently as a client and as a consultant, it is possible for a client to choose itself as a consultant. [9], [11].

Pseudo code of Consultant Guided Search algorithm

```

1 procedure CGSMetaheuristic ( )
2   create the set  $\mathcal{P}$  of virtual persons
3   foreach  $p \in \mathcal{P}$  do
4     setSabbaticalMode (p)
5   end foreach
6   while (termination condition not met) do
7     foreach  $p \in \mathcal{P}$  do
8       if actionMode[p] = sabbatical then
9         currStrategy[p]  $\leftarrow$  constructStrategy (p)
10      else
11        currCons[p]  $\leftarrow$  chooseConsultant (p)
12        if currCons[p]  $\neq$  null then
13          currSol[p]  $\leftarrow$  constructSolution (p,
currCons[p])
14        end if
15      end if
16    end foreach
17    applyLocalOptimization ( ) // optional
18    foreach  $p \in \mathcal{P}$  do
19      if actionMode[p] = sabbatical then
20        if currStrategy[p] better than bestStrategy[p]
then
21          bestStrategy[p]  $\leftarrow$  currStrategy[p]
22        end if
23      else
24         $c \leftarrow$  currCons[p]
25        if  $c \neq$  null and currSol[p] is better than all solutions
found by a client of c since last sabbatical
then
27          successCount[c]  $\leftarrow$  successCount[c] + 1
28          strategy[c]  $\leftarrow$  adjustStrategy(c,
currSol[p])
29        end if
30      end if
31    end foreach
32    updateReputations ( )
33    updateActionModes ( )
34  end while
35 end procedure

```

The above pseudo code shows the actions taken place for a virtual person in sabbatical mode or in normal mode. When a virtual person is set to sabbatical mode, it will take sabbatical leave and stop offering advice to its client. After sabbatical duration, a virtual person is set to normal mode and then it starts offering suggestions.

Consultant strategy are processed according to the random pheromone generation of ant, here (r,u) represents the edge points, and $\tau(r,u)$ stands for the pheromone on edge(r,u). $\eta(r,u)$ is the desirability of edge (r,u), which is usually defined as the inverse of the length of edge (r,u). q is a random number uniformly distributed in [0,1], q_0 is a user-defined parameter with $(0 \leq q_0 \leq 1)$, β is the parameter controlling the relative importance of the desirability. $J(r)$ is the set of edges available at decision point r. S is a random variable selected according to the probability distribution given below [14]. Based on the ant random generation consultants are generating random strategy solutions to the clients.

$$P(r,s) = \begin{cases} \frac{[\tau(r,s)] \cdot [\eta(r,s)]^\beta}{\sum_{u \in J(r)} [\tau(r,u)] \cdot [\eta(r,u)]^\beta}, & \text{if } s \in J(r) \end{cases} \longrightarrow (3.1)$$

The main aim of the proposed CGS technique is to discover the optimal value of JSSP. Scheduling is the allocation of scarce resources to task over time. The main objective is to find the assignment of all resources to all tasks, such that the local total time of the scheduling is to be minimized. In this proposed CGS algorithm, at the beginning of each iteration a client chooses consultants that will advice it during the construction of the current feasible assignment cost. A clients achieves a success, if it constructs scheduling time that is better than all scheduling time found up to that point by any client guided by the same consultant. In this case, the consultant's reputation will be incremented and he will also receive a supplementary bonus. On the other hand

consultant's reputation fades over at each iteration time. The consultant's reputation fading is as given by formula (3.2):

$$reputation_k = reputation_k (1-r) \longrightarrow (3.2)$$

Where, the parameter 'r' represents the reputation fading rates, 'k' indicates which consultant reputation wants to fade over. To keep its reputation, a consultant needs that its clients continuously achieve success. For each consultant, the algorithm keeps track of the best scheduling time obtained by any client working under its guidance. Based on this best schedule of all resources to all tasks over time, the CGS algorithm maintains a ranking of the consultants. Besides reputation, another factor contributing to the choice of a consultant is clients' personal preference. In this CGS algorithm the personal preference for a consultant is determined by the cost of it's advertising for scheduling. Together with reputations, it gives the suitability of consultants by formula (3.3):

$$Suitability_k = \frac{reputation_k}{\beta + time_k - time_{bsf}} \longrightarrow (3.3)$$

Where, the parameter "β" indicates the influence of the personal preference, "time_k" is the time of the schedule advertised by consultant k, "time_{bsf}" is the cost of the best so far schedule. The probability to choose consultant k is:

$$P_k = \frac{suitability_k}{\sum_{c \in C} suitability_c} \longrightarrow (3.4)$$

Where 'c' is the set of all available consultants. A Client is allowed to choose itself as a consultant, because the probabilities given by formula (3.3) do not depend on the client making the choice.

IV. Discussion of Results

Benchmark Problems are gathered to test the robustness of the algorithm. In the domain of JSSP, there are different well-known JSSP benchmarks, which are often utilized to test the algorithms. In this research work, the JSSP benchmark instances from ORLIB [21] are used to test the efficiency of Consultant Guided Search Algorithm (CGS). Lawrence (la), Adams, Balas and Zawack (abz), Applegate and Cook (orb) benchmark JSSP problems were solved in this research work using the new nature inspired metaheuristic technique. Obtained results were compared with both lower bound and upper bound result [22]. Many benchmark instances are available to solve the JSSP. Depending upon the instance size the CGS algorithm generates the optimal result with minimization of makespan.

TABLE 4.1: IMPLEMENTATION RESULT FOR 10*5 JSSP INSTANCE

Instance Name	Instance Size	Lower Bound	Upper Bound	Optimal solution obtained by CGS
la01	10*5	666	830	713
la02		635	844	757
la03		588	773	682
la04		537	839	669
la05		593	677	593
abz5	10*10	868	1370	1347
abz6		742	1071	998
la16		717	1230	1010
la18		663	1128	914
la19		685	1046	944
la20		756	1210	972
ft10		655	1262	1044

TABLE 4.2: IMPLEMENTATION RESULT FOR 10*10 JSSP INSTANCE

Instance Name	Instance Size	Lower Bound	Upper Bound	Optimal solution obtained by CGS
orb01	10*10	695	1456	1255
orb02		671	1157	1045
orb03		648	1297	1229
orb04		759	1356	1171
orb05		630	1116	800
orb06		715	1356	990
orb07		286	502	350
orb08		585	1139	969
orb09		661	1376	1027
orb10		681	1284	1145

Table 4.1 shows the difference between CGS optimal, lower bound and upper bound values obtained from the literature. The CGS gave optimal solutions for the JSSP. The instances are la01, la02, la03, la04 and la05 of size (10*5), abz5, abz6, la16, la18, la19, la20 and ft10 of size (10*10), for la05 instance, CGS could obtain the same lower bound solution. In case of other instances, CGS obtained feasible solutions, which lie between the lower bound and upper bound. Table 4.2 shows the difference between CGS optimal, lower bound and upper bound

values obtained from the literature. The instances are orb01, orb02, orb03, orb04, orb06, orb07, orb08, orb09 and orb10 of (10*10) size. CGS obtained feasible solutions, which lie between the lower bound and upper bound.

TABLE 4.3: IMPLEMENTATION RESULT FOR 15*5 AND 15*10 JSSP INSTANCE

Instance Name	Instance Size	Lower Bound	Upper Bound	Optimal solution obtained by CGS
la06	15*5	926	1078	926
la08		863	966	897
la09		951	1193	956
la10		958	1012	958
la21	15*10	935	1417	1139
la24		857	1356	1092
la25		864	1383	1150
la23		1032	1314	1167
la11	20*5	1222	1577	1222
la12		1039	1286	1058
la13		1150	1337	1152
la14		1292	1497	1292
la15		1207	1442	1310

TABLE 4.4: IMPLEMENTATION RESULT FOR SWV (20*5, 20*15), YN (20*20) JSSP INSTANCE

Instance Name	Instance Size	Best-so-for	Optimal solution obtained by CGS
swv01	20*5	1407	1776
swv02		1475	1815
swv03		1398	1805
swv06	20*15	1675	2168
swv07		1594	2098
swv08		1755	2329
swv09		1656	2233
swv10		1743	2210
yn01	20*20	884	1020
yn02		904	1052
yn03		892	1084

Table 4.3 shows the difference between CGS optimal, lower bound and upper bound values obtained from the literature. The instances are la06, la08, la09 and la10 of size (15*5), la21, la24, la23 and la25 of size (15*10), la11, la12, la13, la14 and la15 of size (20*5) for instances la06, la10 of size (15*5), for la11, la14 of size (20*5), CGS could obtain the same lower bound solution. In case of other instances, CGS obtained feasible solutions, which lie between the lower bound and upper bound. In table 4.4 the instances are swv01, swv02 and swv03 of size (20*5), swv06, swv07, swv08, swv09 and swv10 of size (20*15), yn01, yn02 and yn03 of size (20*20), shows difference between CGS optimal solution and best-so-for solutions. These instances are obtained the solutions is near to the best-so-for solutions.

V. Conclusion & Future Work

In this paper a new Consultant-Guided Search (CGS) metaheuristic algorithm is applied to the Job Shop Scheduling Problem (JSSP) to solve problem in an efficient way. JSSP is one of the NP-Hard problems in combinatorial optimization environment. JSSP is a challenging problem in production and manufacturing environment. JSSP deals with assignment of jobs to machines with operation constraints and machine constraints. The JSSP benchmark instances were tested by using this CGS algorithm. It was found that CGS metaheuristic was effective in solving JSSP instances and it could obtain better solutions for most of the problem cases considered in this research. It was observed that CGS algorithm has obtained lower bound (minimization of makespan) and best-so-for solutions for almost all the instances taken for study. The Consultant-Guided Search (CGS) algorithm for Job Shop Scheduling Problem (JSSP) can also be extended by using some hybrid heuristic algorithms for obtaining better solutions compared to current solutions. CGS can be improved by making each consultant to opt for a different search technique. Implementation of CGS algorithm for large size instance can also be done in future.

VI. References

1. Ahmed Tariq Sadiq and Kanar Shukr Mohammed, "Improved Scatter Search for Job Shop Scheduling Problem", International Journal of Research and Reviews in Soft and Intelligent Computing (IJRRSIC) Vol. 2, No. 1, ISSN: 2046-6412 © Science Academy Publisher, 2012.
2. Apinanthana Udomsakdigool, Voratas Khachitvichyanukul, "Ant Colony Algorithm for Multi-Criteria Job Shop Scheduling to Minimize Makespan, Mean Flow Time and Mean Tardiness", International Journal of Management Science and Engineering management, 6(2):117-123, 2011.
3. C. Blum and A. Roli, "Metaheuristics in combinatorial optimization: Overview and conceptual comparison", ACM Computing Surveys, 35(3):268-308, 2003.
4. Christian Blum, Jakob Puchinger, G'unter Raidl, Andrea Roli, "A Brief Survey on Hybrid Metaheuristics", 2012.
5. Elisa Valentina Onet and Ecaterina Vladu, "Nature Inspired Algorithms and Artificial Intelligence, Journal of Computer Science", 2005.
6. Eugeniusz Nowicki and Czesl Aw smutnicki, "An Advanced Tabu Search Algorithm for the Job Shop Problem", Journal of Scheduling, Volume 8 Issue 2, Pages 145 - 159, 2005.
7. F. Glover and G. Kochenberger, "Handbook of Metaheuristics", Vol 57 of International Series in Operations Research & Management Science. Kluwer Academic Publishers, 2003.
8. Hashni, T and Amudha, T., "Solving Quadratic Assignment Problems using A Hybrid Nature-Inspired Technique", International Journal of Current Research Vol. 4, Issue, 02, pp.216-221, February, 2012.
9. Iordache.S., "Consultant-Guided Search Combined with Local Search for the Quadratic Assignment Problem", GECCO

- (Companion), Springer-Verlag Berlin Heidelberg, 2010.
10. Iordache.S., “*Consultant-Guided Search Combined with Local Search for the Traveling Salesman Problem*”, GECCO (Companion), Springer-Verlag Berlin Heidelberg, 2010.
11. Iordache.S., “*Consultant-Guided Search - A New Metaheuristic for Combinatorial Optimization Problems*”, In Proceedings of the 2010 Genetic and Evolutionary Computation Conference (GECCO 2010). ACM Press, New York (2010).
12. James Montgomery, Carole Fayad1 and Sanja Petrovic., “*Solution Representation for Job Shop Scheduling Problems in Ant Colony Optimisation*”, Springer-Verlag Berlin, 2006.
13. Johann Hurink, Bernd Jurisch, and Monika Thole., “*Tabu Search for the Job-Shop Scheduling Problem Eith Multi-Purpose Machines*”, OR Spektrum 15:205-215,1994.
14. Jun Zhang, Xiaomin Hu, X. Tan, J.H. Zhong and Q. Huang., ” *Implementation of an Ant Colony Optimization Technique for Job Shop Scheduling Problem*”, Transactions of the Institute of Measurement and Control, 34:546-556, 2006.
15. Kaur B and Mittal U., “*Optimization of TSP Using Genetic Algorithm*”, advances in computational sciences and technology, 2010.
16. Kennedy, Online document site address “<http://dsp.jpl.nasa.gov/members/payman/swarm/kennedy95-ijcnn.pdf>”1995.
17. Lewis Ntamo And Suvrajeet Sen,” *The Million-Variable “March” for Stochastic Combinatorial Optimization*”, Journal of Global Optimization 32: 385–400 © Springer 2005.
18. Li-Pei Wong, Chin Soon Chong., ”*Bee Colony Optimization Algorithm with Big Valley Landscape Exploitation for Job Shop Scheduling Problems*”, Proceedings of the Winter Simulation Conference,S. J. Mason, R. Hill, L. Moench, and O. Rose, eds,2008.
19. Mahanim Omar, Adam Baharum, Yahya Abu Hasan,“*A Job-Shop Scheduling Problem (JSSP) Using Genetic Algorithm (GA)*”,Proceesings of the 2nd IMT-GT regional Conference on Mathematics,Statistics and Applications Universi Sains Malaysia,penang, 2006.
20. Mauricio G.C. Resende, Panos M. Pardalos, And Sandra Duni Eks_Io ~Glu,”*Parallel Metaheuristics For Combinatorial Optimization*”, Natal (RN) Brazil, October, 1999.
21. Online document for “JSS benchmark instances”cited at <http://people.brunel.ac.uk/~mastjjb/jeb/orlib/jobshopinfo.html>.
22. Online document for “JSS benchmark results“ cited at <http://bach.istc.kobe-u.ac.jp/csp2sat/jss/>, 2007.
23. Parviz Fattahi, Mohammad Saidi Mehrabad, Mir B. Aryanezhad.,”*An Algorithm for Multi-Objective Job Shop Scheduling Problem*”, Journal of Industrial Engineering International, Vol. 2, No. 3, 43-53 August 2006.
24. Petrică C. Pop, Serban Iordache.,” *A Hybrid Heuristic Approach for Solving the Generalized Traveling Salesman Problem*”, GECCO’11, Dublin, Ireland,Copyright ACM 978-1-4503-0557-0, 2011.
25. Philip N.Klein, Neal E.Young”*Approximation algorithms for NP-hard optimization problems*”, chapter 34, Algorithms and Theory of Computation Handbook © 1999 CRC Press.
26. Qinghai Bai,“*Analysis of Particle Swarm Optimization Algorithm*”,Computer and information science,Vol 3,No1, 2010.
27. Shahrzad Nikghadam, Hamed Kharrati Shishvan, Sohrab Khanmohammadi., ”*Minimizing earliness and tardiness cost in job-shop scheduling problems considering job due dates*”proceedings of international conference on Advances in industrial and production engineering, ©AMAE,2011.